REMARKS

Upon entry of this amendment, claims 1-5, 8-13, 19-22, 30, 32, 33, 35, 36, 40, 42, 43 and 48-54 are all the claims pending in the application. Claims 16-18, 23-28, 37-39 and 44-47 have been canceled by this amendment.

Applicants note that in the Office Action dated March 26, 2007, the Examiner indicated on the Office Action Summary Form that "some" of the certified copies of the foreign priority documents have been received. In this regard, Applicants note that the present application claims priority to only one foreign application, and that the certified copy of this application has been received by the PTO. Accordingly, Applicants kindly request that the Examiner confirm in the next Office paper that "all" of the certified copies of the foreign priority documents have been received.

I. Allowable Subject Matter

Applicants note that the Examiner has not rejected claims 30, 32, 33, 40, 42 and 43 in the Office Action. Thus, Applicants presume that the Examiner considers these claims to be allowable, even though no explicit indication was made by the Examiner in the Office Action to this effect.

II. Claim Rejections under 35 U.S.C. § 101

Claims 5, 13, 23, 28 and 35 have been rejected under 35 U.S.C. § 101 as being directed to non-statutory subject matter. In this regard, the Examiner has indicated in the Office Action that these claims include only "non-functional descriptive material" because a source code program cannot be executed by a computer. Applicants respectfully disagree with the Examiner's position.

In particular, Applicants note that the MPEP indicates that "functional descriptive material" consists of data structures and computer programs which impart functionality when employed as a computer component (see MPEP 2106.01). Thus, with respect to source programs, even though such programs may not be able to be directly executed by a computer,

Applicants note that direct execution of a data structure or computer program is not required in order to be considered "functional descriptive material."

Instead, as noted above, in order to be considered "functional descriptive material", the MPEP indicates that the computer program must impart functionality when employed as a computer component. In this regard, Applicants submit that the source program recited in claims 5, 13, 23, 28 and 35 is clearly imparts functionality when employed as a computer component because the source program directs the compiler, which is controlled by a computer, to perform certain functions.

For example, regarding claim 5, Applicants note that this claim recites, in part, a computer-readable medium on which a source program is recorded, wherein the source program includes at least one of descriptions for directing a compiler that translates the source program into a machine language (1) not to allocate a specific array data to a global memory region and (2) to allocate the specific array data to the global memory region.

With respect to the above-noted features, Applicants respectfully submit that such features clearly impart functionality, namely, the function of directing a complier whether or not to allocate a specific array to the global memory region. Further, because the features in claim 5 direct a compiler to perform certain functions, and the compiler is controlled by a computer, Applicants submit that the source program of claim 5 is clearly employed as a computer component.

In view of the foregoing, Applicants respectfully submit that claim 5 is directed to "functional descriptive material" based on the guidelines set forth in MPEP 2106.01 because the source program described therein imparts functionality when employed as a computer component. Accordingly, Applicants kindly request that the rejection under 35 U.S.C. 101 be reconsidered and withdrawn.

Regarding claim 13, Applicants note that claim 13 recites, in part, a computer readable recording medium on which a source program is recorded, wherein the source program includes at least one of descriptions for directing a compiler that translates the source program into a machine language program (1) not to perform the optimization by software pipelining of a

specific loop processing, (2) to perform optimization that removes a prolog portion and an epilog portion by software pipelining of the specific loop processing, and (3) to perform optimization that does not remove the prolog portion and the epilog portion by software pipelining of the specific loop processing.

For at least similar reasons as discussed above with respect to claim 5, Applicants respectfully submit that claim 13 is directed to statutory subject matter under 35 U.S.C. 101. Accordingly, Applicants kindly request that the rejection be reconsidered and withdrawn.

Regarding claim 23, Applicants note that this claim recites, in part, a computer-readable recording medium on which a source program is recorded, wherein the source program includes at least one of descriptions for directing a compiler that translates the source program into a machine language program (1) to perform the optimization by loop unrolling of a specific loop processing, (2) not to perform the optimization by loop unrolling of a specific loop processing, and (3) to require a guarantee on the number of iterations of a specific loop processing.

For at least similar reasons as discussed above with respect to claim 5, Applicants respectfully submit that claim 23 is directed to statutory subject matter under 35 U.S.C. 101. Accordingly, Applicants kindly request that the rejection be reconsidered and withdrawn.

Regarding claim 28, Applicants note that this claim recites, in part, a computer-readable recording medium on which a source program is recorded, wherein the source program includes at least one of descriptions for directing a compiler that translates the source program into a machine language program (1) to make an "if" conversion to a specific "if" structure sentence and (2) not to make the "if" conversion to the specific "if" structure sentence.

For at least similar reasons as discussed above with respect to claim 5, Applicants respectfully submit that claim 28 is directed to statutory subject matter under 35 U.S.C. 101. Accordingly, Applicants kindly request that the rejection be reconsidered and withdrawn.

Regarding claim 35, Applicants note that this claim recites, in part, a computer-readable recording medium on which a source program is recorded, wherein the source program includes at least one of descriptions for directing a compiler that translates the source program into a machine language program (1) to require a guarantee on alignment of data that a pointer variable

of argument shown by the name of a specific variable indicates and (2) to require a guarantee on alignment of data that a local pointer variable shown by the name of a specific variable indicates.

For at least similar reasons as discussed above with respect to claim 5, Applicants respectfully submit that claim 35 is directed to statutory subject matter under 35 U.S.C. 101. Accordingly, Applicants kindly request that the rejection be reconsidered and withdrawn.

III. Claim Rejections under 35 U.S.C. § 103(a)

A. Claims 1-5 and 36 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Stallman ("Using and Porting the GNU Compiler Collection for GCC 3.1") in view of Nakamura ("Architecture and Compiler Co-Optimization for High Performance Computing"), and further in view of Popovic ("A C Compiler Design Concept Used for MAS Family of Digital Signal Processors").

Claim 1 recites that the global memory region is specified by a head address and a displacement, wherein the head address is indicated by a value stored in a register, wherein the displacement is within a range of the global memory region that can be accessed by one instruction, and wherein the range is determined based on a type and a size of an object.

In the Office Action, the Examiner has recognized that Stallman and Nakamura do not disclose or suggest such the above-noted features. The Examiner, however, has applied Popovic and has taken the position that this reference cures the deficiencies of Stallman and Nakamura. Applicants respectfully disagree.

In particular, with respect to Popovic, Applicants note that this reference merely discloses a pragma directive called mem_alloc which enables programmers to manually allocate memory resources (i.e., to place variables at specific addresses) (see page 607, section II).

In this regard, Applicants respectfully submit that the ability to place variables at specific addresses, as disclosed in Popovic, does not in any way whatsoever correspond to the above-noted features in claim 1 of a head address being indicated by a value stored in a register, and a displacement that is within a range of the global memory region that can be accessed by one instruction, wherein the range is determined based on a type and a size of an object.

For example, with respect to the above-noted feature drawn to the head address being indicated by a value stored in a register, while the Examiner has pointed to the syntax of the mem_alloc directive as including "address", Applicants respectfully submit that the mere indication that "address" is part of the syntax of the directive does not mean that that the head address is indicated by a value stored in a register.

In addition, with respect to the above-noted feature of the displacement being within a range of the global memory region that can be accessed by one instruction, Applicants note that the Examiner has not cited to any portion of Popovic which allegedly discloses this feature. In this regard, Applicants submit that the directive mem_alloc does not in any way whatsoever indicate that a displacement is within a range of the global memory region that can be accessed by one instruction.

Further, with respect to the above-noted feature which indicates that the range can be determined based on a type and a size of an object, Applicants note that the Examiner has not addressed this feature in the Office Action. In this regard, Applicants submit that Popovic includes absolutely no description or suggestion regarding a range which is determined based on a type and a size of an object.

In view of the foregoing, Applicants respectfully submit that the cited prior art does not teach, suggest or otherwise render obvious at least the above-noted features recited in claim 1. Accordingly, Applicants submit that claim 1 is patentable over the cited prior art, an indication of which is kindly requested. Claims 2-4 depend from claim 1 and are therefore considered patentable at least by virtue of their dependency.

Regarding claims 5 and 36, Applicants note that each of these claims recites that the global memory region is specified by a head address and a displacement, wherein the head address is indicated by a valued stored in a register, wherein the displacement is within a range of the global memory region that can be accessed by one instruction, and wherein the range is determined based on a type and a size of an object.

For at least similar reasons as discussed above with respect to claim 1, Applicants respectfully submit that the cited prior art references do not teach, suggest or otherwise render

obvious such features. Accordingly, Applicants submit that claims 5 and 36 are patentable over the cited prior art, an indication of which is kindly requested.

B. Claims 8 and 52 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Stallman in view of Sun (Sun Workshop Compiler c. 4.2 document: C user's guide).

Claim 8 recites that the optimization unit performs optimization on software pipelining following a directive when the directive acquisition unit acquires the directive on optimization by software pipelining, wherein the directive acquisition unit detects a directive for not performing the optimization by software pipelining of a specific loop processing in the source program, and wherein the optimization unit restrains the optimization by software pipelining of loop processing that is an object of the directive detected by the directive acquisition unit.

In the Office Action, the Examiner has recognized that Stallman does not disclose or suggest such the above-noted features. The Examiner, however, has applied Sun and has taken the position that this reference cures the deficiencies of Stallman. Applicants respectfully disagree.

In particular, with respect to Sun, Applicants note that this reference discloses the use of a pragma that specifies that a loop is pipelineable, and that the value of a loop rolling dependency is used to determine what kind of pipelining is to be performed (see page 12).

In this regard, Applicants respectfully submit that while Sun disclose the ability to specify that a loop is pipelineable and determine what kind of pipelining is to be performed, that Sun does not disclose or suggest the software pipelining itself. As such, Applicants respectfully submit that Sun does not disclose or suggest the above-noted feature of an optimization unit that performs optimization on software pipelining following a directive when the directive acquisition unit acquires the directive on optimization by software pipelining.

In addition, with respect to the above-noted features which indicate that the directive acquisition unit detects a directive for not performing the optimization by software pipelining of a specific loop processing in the source program, and wherein the optimization unit restrains the optimization by software pipelining of loop processing that is an object of the directive detected

by the directive acquisition unit, Applicants note that because Sun does not disclose or suggest software pipelining itself, as described above, that Sun does not disclose or suggest the above-noted features.

Moreover, Applicants note that in the Office Action, the Examiner has indicated that it would have been obvious that if a directive does not specify pipelining, that pipelining will not be performed (see Office Action at page 10). In response, Applicants note that the Examiner has misinterpreted the claim language.

In particular, Applicants note that claim 8 does not indicate that optimization on software pipelining does not take place when a directive is received that specifies something other than software pipelining. Instead, as shown above, claim 8 clearly indicates that a directive is detected for not performing the optimization by software pipelining of a specific loop processing in the source program. Applicants respectfully submit that Sun clearly does not disclose, suggest or otherwise render obvious such a feature.

In view of the foregoing, Applicants respectfully submit that the cited prior art references do not teach, suggest or otherwise render obvious the above-noted features recited in claim 8. Accordingly, Applicants submit that claim 8 is patentable over the cited prior art, an indication of which is kindly requested. Claim 52 depends from claim 8 and is therefore considered patentable at least by virtue of its dependency.

C. Claims 9-13, 37, 53 and 54 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Stallman in view of Sun, and further in view of Granston (US 6,892,380).

Regarding claim 9, Applicants note that this claim recites that the directive acquisition unit detects a directive for performing the optimization by software pipelining that removes a prolog portion and an epilog portion.

In the Office Action, the Examiner has recognized that Stallman and Sun do not teach or suggest the above-noted features. The Examiner, however, has applied Grantson and has taken the position that this reference cures the deficiencies of Stallman and Sun. Applicants respectfully disagree.

In particular, regarding Grantson, Applicants note that this reference is directed to a method of software pipelining which provides the ability to omit the epilog (see col. 4, lines 19-26). Thus, while Grantson discloses that the epilog can be removed, Applicants note that Grantson does not disclose that a directive is detected for performing optimization by software pipelining that removes a prolog portion and an epilog portion of a specific loop processing in the source program.

In this regard, it is noted that the Examiner has not addressed in the Office Action the above-noted feature of the "prolog portion" being removed.

Further, with respect to the above-noted feature of a directive for performing the optimization of a specific loop processing, the Examiner has indicated in the Office Action that it is well know in the computer art that the compilation directive can be added to selectively optimize a specific loop (see Office Action at page 11). In view of the Examiner's comments, it is clear that the Examiner is taking Official Notice with respect to such a feature.

Accordingly, if the Examiner maintains the rejection, Applicants request that the Examiner cite to a reference in support of this position as required by MPEP §2144.03.

In this regard, even if the Examiner cites to a reference in support of such a position, Applicants submit that it would not have been obvious to one of ordinary skill in the art to modify Stallman to provide such a feature because Stallman is explicitly directed to an optimization which is performed on all loops whose number of iterations can be determined.

In view of the foregoing, Applicants respectfully submit that the cited prior art references do not disclose, suggest or otherwise render obvious the above-noted combination of features recited in claim 9. Accordingly, Applicants submit that claim 9 is patentable over the cited prior art, an indication of which is kindly requested. Claim 53 depends from claim 9 and is therefore considered patentable at least by virtue of its dependency.

Regarding claim 10, Applicants note that this claim recites that the directive acquisition unit detects a directive for performing the optimization by software pipelining that does not remove the prolog portion and the epilog portion of a specific loop processing in the source program.

In the Office Action, the Examiner has rejected claim 10 based on the same rationale as claim 9. As is clear from the language in claim 10, however, claim 10 recites that a directive is detected for performing the optimization by software pipelining does not remove the prolog portion and the epilog portion. In contrast to this claim language, the Examiner has relied on Granston for the teaching of the removal of an epilog portion.

As such, it is clear that the Examiner's rejection of claim 10 is improper because the claim language has not been properly addressed. Therefore, Applicants submit that claim 10 is patentable over the cited prior art, an indication of which is kindly requested.

Further, for at least similar reasons as discussed above with respect to claim 9, Applicants respectfully submit that there would be no reason to modify the cited prior art references by providing a compilation directive that can selectively optimize a specific loop. As discussed above, if the Examiner maintains that such a feature is well known in the art, Applicants request that the Examiner cite a reference in support of such a position in accordance with MPEP 2144.03, and if such a reference is cited, Applicants request that the Examiner explain with particularity why one of ordinary skill in the art would make such a modification.

In view of the foregoing, Applicants respectfully submit that the cited prior art references do not teach, suggest or otherwise render obvious all of the features recited in claim 10. Accordingly, Applicants submit that claim 10 is patentable over the cited prior art, an indication of which is kindly requested. Claim 54 depends from claim 10 and is therefore considered patentable at least by virtue of its dependency.

Regarding claim 11, Applicants note that this claim recites that the directive acquisition unit detects a designation of the number of iterations of specific loop processing in the source program, wherein the optimization unit performs optimization of loop processing that is an object of the designation detected by the directive acquisition unit based on the designated number of iterations.

In the Office Action, the Examiner has recognized that Stallman and Sun do not teach or suggest the above-noted features. The Examiner, however, has applied Grantson and has taken

the position that this reference cures the deficiencies of Stallman and Sun. Applicants respectfully disagree.

In particular, regarding Grantson, Applicants note that this reference is directed to a method of dynamic software pipelining which describes that a typical FOR loop, when the loop begins, "n" represents the number of desired iterations. In Granston, it is noted that the dynamic software pipelining is performed by a processor, and therefore, while "n" may represent a desired number of iterations, this actual value of iterations is not fixed. In contrast, Applicants note that claim 11 is directed to static software pipelining performed by a computer.

As such, Applicants respectfully submit that while Granston discloses that "n" represents a desired number of iterations, that such disclosure does not in any way suggest the ability to detect a designation of the number of iterations of specific loop processing in the source program, wherein an optimization unit performs optimization of loop processing that is an object of the designation detected by the directive acquisition unit based on the designated number of iterations.

In view of the foregoing, Applicants respectfully submit that the cited prior art references do not teach, suggest or otherwise render obvious the above-noted features recited in claim 11. Accordingly, Applicants submit that claim 11 is patentable over the cited prior art, an indication of which is kindly requested. Claim 12 depends from claim 11 and is therefore considered patentable at least by virtue of its dependency.

Regarding claim 13, Applicants note that this claim recites that a source program includes at least one of descriptions for directing a compiler that translates the source program into a machine language program (1) not to perform the optimization by software pipelining of a specific loop processing, (2) to perform optimization that removes a prolog portion and an epilog portion by software pipelining of the specific loop processing, and (3) to perform optimization that does not remove the prolog portion and the epilog portion by software pipelining of the specific loop processing.

For at least similar reasons as discussed above with respect to claims 8-10, Applicants respectfully submit that the cited prior art references do not teach, suggest or otherwise render

obvious such a combination of features. Accordingly, Applicants submit that claim 13 is patentable over the cited prior art, an indication of which is kindly requested.

D. Claims 16, 17, 38, 44 and 45 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Stallman in view of PGI (PGI Workstation User's Guide-9 Optimization Directive and Pragmas). As noted above, claims 16, 17, 38, 44 and 45 have been canceled by this amendment, thereby rendering this rejection moot.

E. Claims 18-23 and 46-51 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Stallman in view of PGI, and further in view of Geva (US 6,539,541).

Regarding claims 18, 46 and 47, Applicants note that these claims have been canceled by this amendment.

Regarding claims 19 and 48, Applicants note that each of these claims recites that the optimization unit restrains generation of an escape code that is needed in the case of the number of iterations being 0 when the minimum number is 1 or more.

In the Office Action, the Examiner has recognized that Stallman and PGI do not teach or suggest the above-noted feature. The Examiner, however, has applied Geva and has taken the position this reference discloses such a feature at col. 10, lines 9-10 (see Office Action at page 18). Applicants respectfully disagree.

In particular, Applicants note that col. 10, lines 9-10 of Geva indicates that when the unrolled loop is a counted loop, that there is no need to test for the exit condition inside the unrolled body. In other words, in Geva, because a counted loop is a loop in which the number of iterations that the loop will execute is determined once execution reaches the loop, it is not necessary to test for the exit condition inside the unrolled body.

Based on the foregoing, Applicants note that while Geva discloses that when a loop is a counted loop, it is not necessary to text for the exit condition inside the rolled body, that this disclosure does not in any way suggest that an optimization unit restrains generation of an escape

code that is needed in the case of the number of iterations being 0 when the minimum number is 1 or more.

In this regard, Applicants note that the Examiner has not addressed the above-noted language in the Office Action drawn to the number of iterations being 0 when the minimum number is 1 or more.

In view of the foregoing, Applicants respectfully submit that the cited prior art references do not teach, suggest or otherwise render obvious the above-noted feature recited in claims 19 and 48. Accordingly, Applicants submit that claims 19 and 48 are patentable over the cited prior art, an indication of which is kindly requested.

Regarding claims 20 and 49, Applicants note that each of these claims recites that the optimization unit performs the optimization by loop unrolling when the minimum number is equivalent to or more than the number of development by the loop unrolling.

In the Office Action, the Examiner has recognized that Stallman and PGI do not teach or suggest the above-noted feature. The Examiner, however, has applied Geva and has taken the position this reference discloses such a feature at col. 10, lines 6-9 (see Office Action at page 18). Applicants respectfully disagree. In particular, Applicants note that col. 10, lines 6-9 of Geva discloses that one type of optimization may be loop unrolling where a loop is unrolled "n" times, such that "n-1" additional copies of the loop body are made.

Based on the foregoing, Applicants note that while Geva discloses that a loop may be unrolled "n" times such that "n-1" additional copies of the loop body are made, that such disclosure does not in any way suggest that optimization by loop unrolling is performed when the minimum number is equivalent to or more than the number of development by the loop unrolling, as recited in claims 20 and 49.

In view of the foregoing, Applicants respectfully submit that the cited prior art references do not teach, suggest or otherwise render obvious the above-noted feature recited in claims 20 and 49. Accordingly, Applicants submit that claims 20 and 49 are patentable over the cited prior art, an indication of which is kindly requested.

Regarding claims 21 and 50, Applicants note that each of these claims recites that the designation of the number of iterations guarantees that the loop processing is iterated only an even number of times.

In the Office Action, the Examiner has recognized that the cited prior art references do not disclose such a feature (see Office Action at page 18). The Examiner, however, has indicated that based on the disclosure in Geva which indicates that the number of iterations can be read from an input file, that it would have been obvious to specify the number of iterations in order to ensure that the number of iterations can be determined at compile time (see Office Action at page 19). Applicants respectfully disagree.

Initially, Applicants point out to the Examiner that Geva indicates that the inability of the compiler to determine the number of iterations can be due to the number of iterations being read by a program from an input file.

Further, with respect to the Examiner's position that it would have been obvious to specify the number of iterations, Applicants note that the mere ability to specify the number of iterations does not render obvious the above-noted feature which indicates that the designation of the number of iterations guarantees that the loop processing is iterated only an even number of times.

In other words, even if it would have been obvious to designate the specific number of iterations, Applicants note that this does not mean that it would have been obvious to designate the number of iterations so as to guarantee that the loop processing is iterated only an even number of times, as recited in claims 12 and 50.

In view of the foregoing, Applicants respectfully submit that the cited prior art references do not teach, suggest or otherwise render obvious the above-noted feature recited in claims 21 and 50. Accordingly, Applicants submit that claims 21 and 50 are patentable over the cited prior art, an indication of which is kindly requested.

Regarding claims 22 and 51, Applicants note that each of these claims recites that the designation of the number of iterations guarantees that the loop processing is iterated only an odd number of times.

For at least similar reasons as discussed above with respect to claims 21 and 50, Applicants submit that the cited prior art references do not teach, suggest or otherwise render obvious such a feature. Accordingly, Applicants submit that claims 22 and 51 are patentable over the cited prior art, an indication of which is kindly requested.

F. Claims 24-28 have been rejected under 35 U.S.C. § 103(a) as being unpatentable Stallman in view of Faraboschi (Faraboschi, The Latest Word in Digital and Media Processing). As noted above, claims 24-28 have been canceled by this amendment, thereby rendering this rejection moot.

IV. Conclusion

In view of the above, reconsideration and allowance of this application are now believed to be in order, and such actions are hereby solicited. If any points remain in issue which the Examiner feels may best be resolved through a personal or telephone interview, the Examiner is kindly requested to contact the undersigned at the telephone number listed below.

Respectfully submitted,

Hajime OGAWA et al.

By: Kenneth W. Fields

Kenneth W. Fields
Registration No. 52,430
Attorney for Applicants

KWF/ra
Washington, D.C. 20006-1021
Telephone (202) 721-8200
Facsimile (202) 721-8250
January 14, 2008